Mar 9

Thursday, March 9, 2017 2:49 PM

March 21: Decide what to do for the project

From homework:

Π^3_2

1) deleting 3rd #

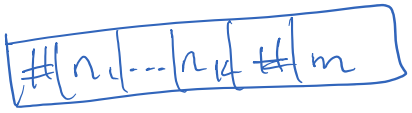
2) Π^2_2

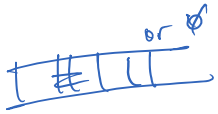
$f(n_1, \dots, n_k) = \mu m \, P(n_1, \dots, n_k, m)$

we have 

we want 

we use TM for checking P:

P starts with 

P ends with 

halt

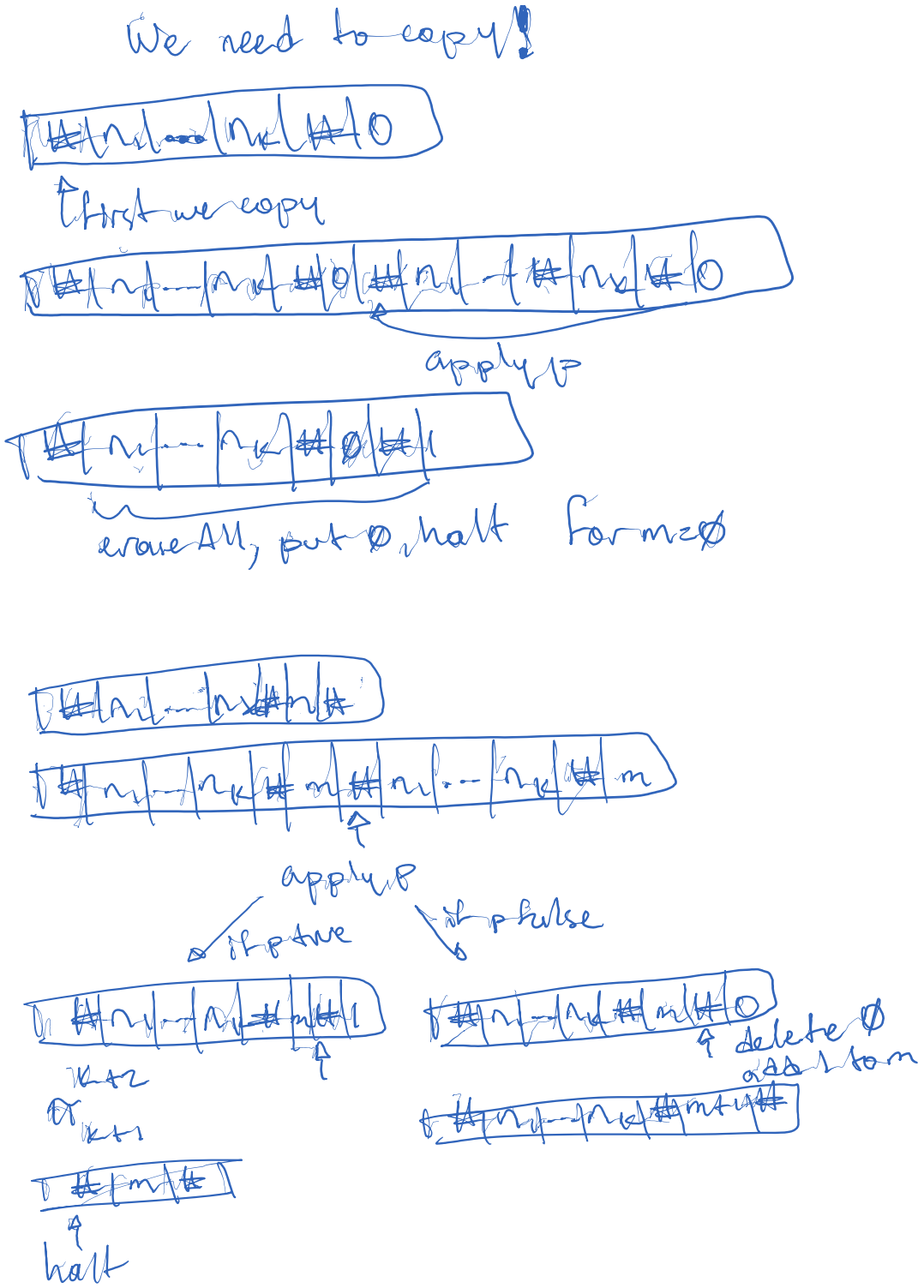
We assume that there is a TM to compute P.
We prove there is a TM for computing $\mu m P(n_1, \dots, n_k, m)$

Brainstorming:

we want to take $m = \emptyset$ check; if yes → return $\emptyset$

if no → try $m = 1$, etc

we can apply TM P and check whether $P(n_1, \dots, n_k, 0)$



Some algorithms are feasible, some are not

Example of feasible algorithms: search, $O(n)$, sorting, $O(n \lg n)$, matrix mult, $O(n^3)$, Dijkstra's algorithm $O(n^3)$ or $O(n^4)$

Example of algorithm not feasible

Propositional satisfiability problem

Given: a propositional expression

$v_1, \dots, v_n, \vee, \&, \neg$

((&& !

$(v_1 \& v_2 \& \neg v_3) \vee (\neg v_1 \& v_2 \& v_3) \vee \dots$

We want: to find the values $v_1, \dots, v_n$ that make this expression true

$P \equiv (v_1 \vee v_2 \vee \neg v_3) \& (\neg v_1 \vee v_2 \vee v_3)$

Why? if we have a code

if (p) {...}

else {...}

How can we solve it?

v_1 - 2 values T, F

v_1, v_2 - 4 values

$v_1, \dots, v_n$ - 2^n combinations

Age of universe

$T = 20 \cdot 10^9$ years $\approx 10^{10}$ years

Δt - time during which light passes through smallest possible particle

$\Delta r \sim \frac{\hbar}{M}$ $\quad \Delta E \cdot \Delta t \simeq \hbar$ Heisenberg

$\Delta t \simeq \frac{\hbar}{\Delta E}$

$1_{day} = 3 \cdot 10^2 day \cdot 24 \cdot 10 \frac{hr}{day} \cdot 36 \cdot 10^3 \frac{sec}{hr} \simeq 3 \cdot 10^7$

$3 \cdot 10^{17}$ sec lifetime of the universe

Size of proton in Armstrong $10^{-10} cm = 10^{-12} m$

$3 \cdot 10^5 Km/sec = 3 \cdot 10^8 m/sec \Rightarrow$ speed of light

$$\frac{10^{-12} m}{3 \cdot 10^8 m/sec} = 3 \cdot 10^{-21} sec$$

$$\frac{3 \cdot 10^{17}}{3 \cdot 10^{-21}} = 10^{39}$$

10^{90} particles

10^{40} steps

$10^{90} \cdot 10^{40} = 10^{130}$

$2^n \quad n = 500$

$2^{10} \simeq 10^3$

$2^{500} = (2^{10})^{50} = (10^3)^{50} \simeq 10^{150}$

#fastest computer running lifetime Universe

Some algorithms are feasible, they are usually polynomial-time. there exists a polynomial $P(n)$ s.t. $\forall$ input x $t_A(x) \leq P(len(x))$

$O(P(len(x)))$

Some algorithms are not feasible

Example: exhaustic search for SAT time 2^n — propositional satisfiability problem

We would like to have a good precise definition of what is feasible and what is not

---

Unfortunately, no one knows how to give such

a definition

The best we have is:

An algorithm is feasible if it is polynomial-time

Why is this not a perfect definition?

* $t_A(x) = 10^{140} \cdot \text{len}(x)$

  from practical view point - not feasible

  (because universe computer will compute 10^{150})

  from viewpoint of definition - perfectly feasible

* $t_A(x) = \exp(10^{-90} \cdot \text{len}(x))$

  from practical viewpoint: feasible

  from definition: not-feasible

  because it will grow faster than polynomial-time

See:

V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, "The notions of feasibility and NP-hardness: brief introduction", Chapter 2 from "Computational complexity and feasibility of data processing and interval computations", Kluwer, Dordrecht, 1997. **pdf file**

From <http://www.cs.utep.edu/vladik/cs5315.17/>

Which problems have feasible algorithms solving them ≡ tractable

which problems don't have feasible algorithm ≡ intractable

we need a good definition of a problem

what is a problem in general?

1) Mathematicians: (They like to prove)

   - Given a statement X
   - Find a proof y of X or ¬X

We have a feasible checking alg $C(x,y)$ that checks whether y is a correct proof

proof has to be of feasible length

$$len(y) \leq P_d(len(x))$$

$C(x,y)$, P_e

$x \xrightarrow{?} y$ s.t. $C(x,y)$

& $len(y) \leq P_e(len(x))$